Program :

Fork

```
#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>


void main() {

    int p,p1,p2,p3;

    p=fork();

    if(p==-1) {

        printf("error");

        exit(1);

    }

    if(p!=0) {

        p1= getpid();

        printf("process id of parent is %d\n",p1);

    }

    else {

        p2=getpid();

        printf("process id of child is %d\n",p2);

    }

}
```

Program :

exec

```
#include<stdio.h>
#include<unistd.h>

void main() {
   int pid;
   pid=fork();
      if(pid==0) {
         printf("exec starts");
         execl("/bin/ls","ls","-l",(char)(0));
         printf("exec didn't work");
      }
      else {
         wait(0);
         printf("parent = ls is completed in child");
      }
}
```

Program :

```
#include<stdio.h>
#include<unistd.h>

void main() {
   printf("Process ID : %d \n",getpid());
}
```

getpid.

Output :

```
Process ID : 198
```

Program :

```
#include<stdio.h>
#include<stdlib.h>
```

exit

```
void main() {
   printf("Hello user!! \nLets start!!\n.\n.\n.\n.\nEnd!!\n");
   exit(EXIT_SUCCESS);
}
```

Output :

```
Hello user!!
Lets start!!
.
.
.
.
End!!
```

Program :

```
#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>


void main() {

    pid_t cpid;

    if(fork()==0) {

        exit(0);

    }

    else {

        cpid=wait(NULL);

    }

    printf("Parent PID : %d\n",getpid());

    printf("Child PID : %d\n",cpid);

}
```

Program :

```
#include<stdio.h>
#include<stdlib.h>
#include<fcntl.h>

int main() {
    int fd1 = open("hi.txt",O_RDONLY);
    if(fd1<0) {
        perror("c1");
        exit(1);
    }
    printf("Opened the fd = %d\n",fd1);
    if(close(fd1)<0) {
        perror("c1");
        exit(1);
    }
    printf("Closed the fd.\n");
}
```

Input :

test.txt

“testing this file”

Program :

```
#include<stdio.h>
#include<sys/stat.h>


void main() {
    struct stat file;
    stat("stat.c",&file);
    printf("ST_Mode : %o \n",file.st_mode);
 }
```

stat

Program :

```
#include <stdio.h>
#include <dirent.h>

int main() {
    DIR *folder;
    struct dirent *entry;
    int files = 0;
    folder = opendir(".");
    if(folder == NULL) {
        perror("Unable to Read Directory");
        return(1);
    }
    while((entry=readdir(folder))) {
        files++;
        printf("File %3d: %s\n",files,entry->d_name);
    }
    closedir(folder);
    return(0);
}
```

opend readdir

Program :

```
#include <stdio.h>
#include <unistd.h>

void create() {
    if(fork() == 0) {
        printf("Child Process Created..!\n");
        printf("Process ID : %d \n\n",getpid());
    }
    else {
        printf("Parent Process Created..!\n");
        printf("Process ID : %d \n\n",getpid());
    }
}

void main() {
    printf("Processing.....\n\n");
    create();
    printf("Process Completed\n\n");
}
```